codee

A Tool to Enhance Correctness, Modernization, Security, Portability, and Optimization in Fortran and C/C++ Software Applications

Manuel Arenaz
manuel.arenaz@codee.com

Latest update

Modern Fortran Development Requirements

Adherence to Modern Standards
Modern Fortran 90/95/2003/2008/2018/2023, Modern C/C++.

Usage of the Latest Tools & Libraries
Optimized Libraries, Compilers.

Modularization of the Code to Favor Maintenance
Encapsulation, Modularization, OO Programming.

Collaboration
DevOps, Version Control Systems, CI/CD, Containers.

Correctness
Static Analyzers, Sanitizers.

Portability
Cross-Compilers.

Security
Cybersecurity Regulations, SAST, SCA, DevSecOps.

codee

codee

From **F77** up to **F2023** | **Fixed-form** and **free-form** | **Standard Fortran** Language and Compiler **Non-Standard Dialects** (GNU, Intel)

Code Formatting

Enforce a uniform source code format to write cleaner, more consistent, and maintainable code effortlessly.

Static Analysis

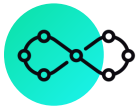
Automatically analyze every line of code to find and fix modernization and optimization opportunities.

Autofix

Automatically generate fixes for opportunities, always under the control of the programmer and preserving 100% code correctness.

Reports

Get a deeper understanding of your code's health with analysis reports.



CI/CD automation

Integrate with CI/CD systems, automatically testing every code change and pull request.

Self-hosting

Execution on the local system, retraining full control of your code and privacy.

CODEE

Formatter

Just Download it and Use it:
https://www.codee.com/codee-formatter/

Most Advanced Source Code Formatter for Fortran:
Highly Customizable, Fast, Reliable and Free

Feature name	Feature description	Codee	fprettify	fortitude	findent
Command-line interface (CLI) Integration	Command-line simple usage, integrable into editors or CI pipelines	✔	✔	✔	✔
Config File Customization	Customization file for code style enforcement with extensive documentation	✔	✘	✔	✘
Partial File Formatting	Format only parts of the code, ideal for IDE selections or git commits	✔	✘	✘	✘
Format Suppression Comments	Suppression of formatting in code sections with comments	✔	✔	✔	✔
Detailed Documentation	Up-to-date detailed documentation with all the options explained	✔	✘	✔	✔
Integration Guides	Step-by-step guides with integration with git, the most used IDEs and CI pipelines	✔	✘	✘	!
Modern Fortran Support	Support up to the latest version of Fortran (2023)	✔	✘	✔	✔
Automatic Line Indentation	Automatic indentation based on code scope	✔	✔	✘	✔
Column Limit	Breaking long lines into multiple smaller ones	✔	✘	!	✘
Operator Spacing Consistency	Consistent spacing around operators and keywords	✔	✔	✘	✘
Uniform Operator Style	Choose between symbolic or literal representation of Fortran operators	✔	✔	✔	✘
Consistent Keyword Casing	Ensure consistent casing of keywords, identifiers and operators	✔	✔	✘	✘
End Statement Style	Ensure joined/separated, with/without names end statements	✔	✘	✔	✔
Double-Colon in Declarations	Add missing double-colon token to variable declaration	✔	✘	✔	✘
Kind Keyword Enforcement	Enforcing the usage of the kind keyword in intrinsic declaration	✔	✘	✘	✘
Split Multiline Statements	Break each statement into a separate line	✔	✘	✘	✘
Remove Superfluous Semicolons	Removes unnecessary semicolons	✔	✘	!	✘
Whitespace Cleanup	Redundant EOL, trailing whitespace and consecutive whitespaces	✔	✔	✔	✔
EOL Normalization	Enforce consistent end-of-line (LF or CRLF)	✔	✘	✘	✘

CODEE

Analyzer

Pricing Plans for an Incremental Adoption in your Organization:
https://www.codee.com/pricing/

Most Advanced Static Analyzer for Fortran Applications, including C/C++ as well

Code	Domain	Metrics with Codee 2025.2.1 (Apr 2025)
CP2K 1.3M lines of code	Quantum chemistry and solid state physics software package	`1361 files (291 with missing deps), 11306 functions, 22399 loops, 1093520 LOCs successfully analyzed (20829 checkers) and 0 non-analyzed files in 1 h 12 m 47 s`
OpenRadioss 1.1M lines of code	Finite element solver for dynamic event analysis	`3519 files, 6692 functions, 39742 loops, 1132227 LOCs successfully analyzed (39412 checkers) and 0 non-analyzed files in 1 h 15 m 0 s`
WRF 960K lines of code	Weather Research and Forecasting	`508 files, 9700 functions, 26246 loops, 949428 LOCs successfully analyzed (75118 checkers) and 0 non-analyzed files in 5 h 17 m 34 s`
ICON 646K lines of code	Weather, climate, and environmental prediction	`1154 files, 11694 functions, 21644 loops, 655567 LOCs successfully analyzed (14788 checkers) and 0 non-analyzed files in 23 m 41 s`
PHASTA 64K lines of code	Parallel Hierarchic Adaptive Stabilized Transient Analysis of compressible and incompressible Navier Stokes equations	`284 files (6 with missing deps), 705 functions, 1408 loops, 63051 LOCs successfully analyzed (2595 checkers) and 0 non-analyzed files in 19 m 35 s`
HYCOM 44K lines of code	Hybrid Coordinate Ocean Model	`50 files, 250 functions, 2107 loops, 45242 LOCs successfully analyzed (1740 checkers) and 0 non-analyzed files in 1 m 11 s`

CODEE

Analyzer

Also useful for Static Analysis of System Software, written in Fortran and C/C++

OpenBLAS
1M lines of code
2K files in Fortran
3K files in C

OpenBLAS is an optimized BLAS library based on GotoBLAS2 1.13 BSD version.

```
5193 files, 7502 functions, 18583 loops, 1035170 LOCs successfully analyzed
(32200 checkers) and 70 non-analyzed files in 27 m 25 s
```

Checker	Category	Priority	AutoFixes	#	Title
PWR079	correctness, portability, security	P27 (L1)		13	Avoid undefined behavior due to uninitialized variables
PWR063	correctness, modern, security	P12 (L1)		715	Avoid using legacy Fortran constructs
PWR068	correctness, modern, security	P9 (L2)		13224	Encapsulate procedures within modules to avoid the risks of calling implicit interfaces
PWR008	correctness, modern, security	P9 (L2)	9	2338	Declare the intent for each procedure argument
PWR007	correctness, modern, security	P9 (L2)	2190	2193	Disable the implicit declaration of variables and procedures
PWR069	correctness, modern, security	P9 (L2)	16	16	Use the keyword only to explicitly state what to import from a module
PWR003	modern, security, other	P6 (L2)		78	Explicitly declare pure functions
PWR018	security, control	P6 (L2)		6	Call to recursive function within a loop inhibits vectorization
PWR071	modern, portability, security	P3 (L3)		7135	Prefer real(kind=kind_value) for declaring consistent floating types
PWR002	correctness, security	P3 (L3)		3376	Declare scalar variables in the smallest possible scope
PWR037	correctness, security	P3 (L3)		13	Potential precision loss in call to mathematical function
PWR073	correctness, modern, security	P3 (L3)		3	Transform common block into a module for better data encapsulation
PWR070	correctness, modern, security, memory	P2 (L3)		2143	Declare array dummy arguments as assumed-shape arrays
PWR028	security, control	P2 (L3)		717	Remove pointer increment preventing performance optimization
PWR030	security, control	P2 (L3)		2	Remove pointer assignment preventing performance optimization for perfectly nested loops
PWR001	correctness, modern, security	P1 (L3)		228	Pass global variables as function arguments
Total			2215	32200	

Codee Enhances the Cray Programming Environment (CPE)

Cray Programming Environment (CPE) is a powerful ecosystem of developer tools

+

Codee provides new tools for Fortran and C/C++

Cray Compiler Environment (CCE)

Performance Analysis Tools

Static Analyzer for Fortran, (and C/C++)

Code Formatter for Fortran

Optimized Libraries

Parallel Programming

Static Application Security Testing (SAST)

Software Composition Analysis (SCA)

TESTIMONIALS

What Users are Saying about Codee...

"Codee is a gem. It's a huge time-saver for junior and senior devs alike. Sure, there's some overlap with tools I've used, but ***most of its detections are unique and valuable.****"*

Matthaios Alexandrakis
PhD, Research Software Engineer at the University of Greenwich

"If we include the time we spent verifying the code's validity after it was modernized, and the time we lost for formatting the Fortran code, ***the Codee approach for modernization saves 25 times more time****."*

Veselin Kolev
PhD, CEO Cluster Operations at Discoverer Supercomputer PetaSC Bulgaria

[CUG2025 keynote]

"Bugs happen"

"Static analysis tools have gotten really good"

*"****static analyzers like clang-tidy make wonders****"*

Michael Zingale
Michael Zingale (Stony Brook University)

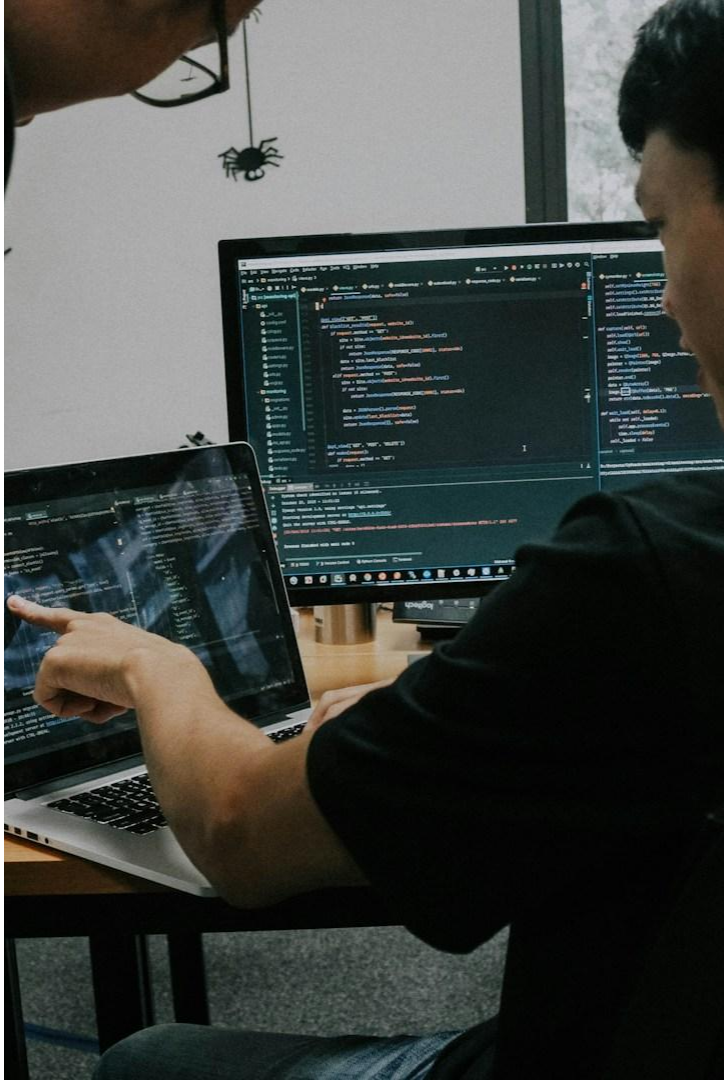
codee

Trusted by leading organizations around the world

SECURITY

Codee Helps Secure Fortran Codes

- Cybersecurity Regulations in US/EU affect HPC
- Compliance is a Must, also for Fortran and C/C++
- Need DevSecOps Tools to Automate Vulnerability Scans
- Codee Analyzer provides Static Analysis for Fortran, including also C/C++
- SAST: Static Application Security Testing
- SCA: Software Composition Analysis



Thank you!

Manuel Arenaz
manuel.arenaz@codee.com

www.codee.com

info@codee.com

Subscribe: codee.com/newsletter/

Spain

codee_com

/codee-com/